DESENVOLVIMENTO DE APPS
- PARTE 1 (ANDROID)

MAPAS & MULTIMÍDIA

ROBERTO RODRIGUES JUNIOR

6

LISTA DE FIGURAS

Figura 6.1 – Primeira tela do App Recursos Multimídia.....6
Figura 6.2 – Recuperando fotos.....12
Figura 6.3 – Utilizando MapView.....16
Figura 6.4 – Informações personalizadas no marcador.....16
Figura 6.5 – Localização do usuário.....21
Figura 6.6 – Tela rotas e pontos de interesse.....27
Figura 6.7 – Ponto de interesse Museu do Catavento Cultural.....33
Figura 6.8 – Utilizando WebView.....37
Figura 6.9 – Reproduzindo áudio.....39
Figura 6.10 – Executando vídeo.....41
Figura 6.11 – Tela com os efeitos programados.....48

LISTA DE CÓDIGOS-FONTE

Código-fonte 6.1 – XML AndroidManifest....7
Código-fonte 6.2 – Dependências a serem adicionadas no build.gradle (Module: app) 8
Código-fonte 6.3 – XML activity_main....9
Código-fonte 6.4 – Classe MainActivity....11
Código-fonte 6.5 – XML activity_recuperando_fotos....14
Código-fonte 6.6 – Classe RecuperandoFotosActivity....15
Código-fonte 6.7 – XML google_maps_api....17
Código-fonte 6.8 – XML activity_utilizando_mapas....17
Código-fonte 6.9 – Classe UtilizandoMapasActivity....20
Código-fonte 6.10 – XML activity_localizacao_usuario....21
Código-fonte 6.11 – Classe LocalizacaoUsuarioActivity....26
Código-fonte 6.12 – XML activity_primeira_rota....27
Código-fonte 6.13 – Classe PrimeiraRotasActivity....32
Código-fonte 6.14 – XML activity_pontos_interesse....33
Código-fonte 6.15 – Classe PontosInteresseActivity....36
Código-fonte 6.16 – XML activity_usando_web_view....37
Código-fonte 6.17 – Classe UsandoWebViewActivity....38
Código-fonte 6.18 – XML activity_sons_audio....40
Código-fonte 6.19 – Classe SonsAudioActivity....40
Código-fonte 6.20 – XML activity_videos_media_player....42
Código-fonte 6.21 – Classe VideosMediaPlayerActivity....44
Código-fonte 6.22 – Animação Fade in....44
Código-fonte 6.23 – Animação Fade out....44
Código-fonte 6.24 – Animação blink....45
Código-fonte 6.25 – Animação move....45
Código-fonte 6.26 – Animação bounce....45
Código-fonte 6.27 – Animação Rotate....46
Código-fonte 6.28 – Animação Slide up....46
Código-fonte 6.29 – Animação Slide down....46
Código-fonte 6.30 – Animação Zoom in....47
Código-fonte 6.31 – Animação Zoom out....47
Código-fonte 6.32 – XML activity_animacao....49
Código-fonte 6.33 – Classe AnimacaoActivity....50

SUMÁRIO

6 MAPAS & MULTIMÍDIA.....5
6.1 Recuperando fotos com ImageView.....11
6.2 Utilizando mapas: MapView.....15
6.3 Localização do usuário com Location API.....20
6.4 Rotas e Pontos de Interesse.....26
6.5 Usando WebView em seus Apps.....36
6.6 Sons: AudioManager.....38
6.7 Vídeos: MediaPlayer.....40
6.8 Animação em views com View Animation.....44
REFERÊNCIAS.....51

6 MAPAS & MULTIMÍDIA

Desenvolver aplicativos mobile para Android com o uso de recursos multimídia pode aperfeiçoar a experiência de uso do aplicativo. Quando ideias surgem e são transformadas em aplicativos é muito importante elaborar telas que explorem o estímulo ao uso do app. Neste capítulo, serão apresentados diversos recursos multimídia que podem ser programados, tais como: o gerenciamento de imagens, mapas, áudio, vídeo e, principalmente, animação.

O objetivo, neste capítulo, é estudar o código apresentado para possibilitar que o implementem em suas rotinas. Observe os resultados oferecidos e perceba que trabalhar com recursos multimídia pode enriquecer a experiência de uso de aplicativos Android.

A Figura “Primeira tela do App Recursos Multimídia” apresenta a tela inicial do aplicativo, o qual tem por objetivo apresentar os diferentes recursos multimídia do Android. O Código-fonte “XML AndroidManifest” consiste no AndroidManifest.xml, que possui as configurações gerais do aplicativo e as respectivas Activity.

Importante observar o uso da tag uses-permission, que habilita o pedido de permissão de recursos do dispositivo Android. Por exemplo, esse aplicativo solicita ao sistema operacional o uso de Internet (INTERNET), localização aproximada (ACCESS_COARSE_LOCATION) e localização precisa (ACCESS_FINE_LOCATION). É importante também observar a tag meta-data com a configuração e chave pública para uso do API do Google Maps. Para a tela inicial, mantenha, inicialmente, a activity MainActivity no AndroidManifest (mantendo as outras comentadas com bloco <!-- -->). Depois, à medida que os exemplos forem abordados, as outras Activity podem ser incluídas.

Figura 6.1 – Primeira tela do App Recursos Multimídia
Fonte: Elaborado pelo autor (2020)

```
      <?xml                version="1.0"                encoding="utf-8"?>
<manifest    xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.android.recursosmultimidia">
    <!--              Recuperando_Fotos_Com_ImageView                -->
    <uses-permission    android:name="android.permission.INTERNET"    />
    <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION"              />
    <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION"                />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
```

```
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <activity                          android:name=".MainActivity">
            <intent-filter>
                <action  android:name="android.intent.action.MAIN"  />
                <category
android:name="android.intent.category.LAUNCHER"                      />
            </intent-filter>
        </activity>

<!--         <activity android:name=".AnimacaoActivity"></activity>-->
<!--              <activity  android:name=".SonsAudioActivity"  />-->
<!--         <activity android:name=".VideosMediaPlayerActivity" />-->
<!--            <activity  android:name=".UsandoWebViewActivity"  />-->
<!--             <activity  android:name=".PrimeiraRotaActivity"  />-->
<!--        <activity android:name=".LocalizacaoUsuarioActivity" />-->
<!--          <activity  android:name=".UtilizandoMapasActivity" />-->
<!--          <activity android:name=".RecuperandoFotosActivity" />-->

        <meta-data
            android:name="com.google.android.geo.API_KEY"
            android:value="AIzaSyDVff6FC6jjSlJiS3xY-rBWJmOA7CWMhQU"  />
    </application>

</manifest>
```

Código-fonte 6.1 – XML AndroidManifest
Fonte: Elaborado pelo autor (2018)

Depois, como mostra o Código-fonte “Dependências a serem adicionadas no build.gradle (Module: app)”, para o funcionamento dos exemplos a seguir, é necessário incluir as dependências do Glide (framework para recuperação de fotos para o ImageView), Google Play Services e JetBrains Anko (facilitador do uso de Features no Android). Sincronize o projeto para que as dependências sejam incluídas.

```
    //-- Recuperando_Fotos_Com_ImageView
    implementation 'com.github.bumptech.glide:glide:3.6.0'
```

```
    //-- Google Play Services
    implementation 'com.google.android.gms:play-
services:11.8.0'
//--Jetbrains
    implementation 'org.jetbrains.anko:anko-sdk15:0.8.2'
    implementation 'com.beust:klaxon:0.30'
```

Código-fonte 6.2 – Dependências a serem adicionadas no build.gradle (Module: app)
Fonte: Elaborado pelo autor (2019)

O Código-fonte “XML activity_main” mostra o layout em XML da MainActivity. Adicione esse Layout e crie as funções apontadas no evento android:onClick utilizado em cada widget. À medida que cada exemplo for abordado, deverão ser implementadas as intents para transição de cada tela.

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="40dp"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <Button
        android:onClick="acessarRecuperacaoFotos"
        android:text="14.1 Recuperando fotos com ImageView"
        android:textSize="14dp"
        android:textAllCaps="false"
        android:gravity="left"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <Button
        android:onClick="acessarMapa"
        android:text="14.2 Utilizando mapas: MapView"
        android:textSize="14dp"
        android:textAllCaps="false"
        android:gravity="left"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <Button
        android:onClick="acessarLocalizacaoDoUsuario"
        android:text="14.3 Localização do usuário com Location API"
        android:textSize="14dp"
        android:textAllCaps="false"
        android:gravity="left"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <Button
        android:onClick="acessarRotas"
```

```
        android:text="14.4 Rotas"
        android:textSize="14dp"
        android:textAllCaps="false"
        android:gravity="left"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <Button
        android:onClick="acessarPontosInteresse"
        android:text="14.5 Pontos Interesse"
        android:textSize="14dp"
        android:textAllCaps="false"
        android:gravity="left"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <Button
        android:onClick="acessarWebView"
        android:text="14.6 Usando WebView em seus Apps"
        android:textSize="14dp"
        android:textAllCaps="false"
        android:gravity="left"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <Button
        android:onClick="acessarAudio"
        android:text="14.7 Sons: AudioManager"
        android:textSize="14dp"
        android:textAllCaps="false"
        android:gravity="left"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <Button
        android:onClick="acessarVideo"
        android:text="14.8 Vídeos: MediaPlayer"
        android:textSize="14dp"
        android:textAllCaps="false"
        android:gravity="left"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <Button
        android:onClick="acessarAnimacao"
        android:text="14.9 Animação em views com View Animation"
        android:textSize="14dp"
        android:textAllCaps="false"
        android:gravity="left"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

</LinearLayout>
```

Código-fonte 6.3 – XML activity_main
Fonte: Elaborado pelo autor (2020)

```
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.content.Intent
import android.view.View

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }

    fun acessarAnimacao(view: View) {
        val intent = Intent(this, AnimacaoActivity::class.java)
        startActivity(intent)
    }

    fun acessarLocalizacaoDoUsuario(view: View) {
        val intent = Intent(this,
LocalizacaoUsuarioActivity::class.java)
        startActivity(intent)
    }

    fun acessarRecuperacaoFotos(view: View) {
        val intent = Intent(this,
RecuperandoFotosActivity::class.java)
        startActivity(intent)
    }

    fun acessarRotas(view: View) {
        val intent = Intent(this, PrimeiraRotaActivity::class.java)
        startActivity(intent)
    }

    fun acessarPontosInteresse(view: View) {
        val intent = Intent(this, PontosInteresseActivity::class.java)
        startActivity(intent)
    }

    fun acessarAudio(view: View) {
        val intent = Intent(this, SonsAudioActivity::class.java)
        startActivity(intent)
    }

    fun acessarWebView(view: View) {
        val intent = Intent(this, UsandoWebViewActivity::class.java)
        startActivity(intent)
    }

    fun acessarMapa(view: View) {
        val intent = Intent(this, UtilizandoMapasActivity::class.java)
        startActivity(intent)
    }

    fun acessarVideo(view: View) {
        val intent = Intent(this,
VideosMediaPlayerActivity::class.java)
        startActivity(intent)
```

```
        }
}
```

Código-fonte 6.4 – Classe MainActivity
Fonte: Elaborado pelo autor (2020)

6.1 Recuperando fotos com ImageView

A utilização de imagens em um aplicativo aperfeiçoa a relação de interesse dos usuários. Desenvolver aplicativos que apresentem fotos, desenhos ou uma iconografia bem elaborada torna-os ainda mais interessantes. Esta seção apresenta como capturar imagens da web e exibi-las sem a necessidade de armazenamento, utilizando o componente Glide. É possível usar esse recurso para estabelecer comunicação com *webservices* de imagens, assim como outras soluções desenvolvidas para Android fazem.

Para o exemplo proposto, crie uma Activity chamada de *RecuperandoFotosActivity*. Depois, utilize o Código-fonte "XML *RecuperandoFotosActivity*" para preenchimento do layout e o Código-fonte "Classe Recuperando_Fotos_Com_ImageView" para preenchimento do código-fonte em Kotlin.

Figura 6.2 – Recuperando fotos
Fonte: Elaborado pelo autor (2020)

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:orientation="vertical"
    tools:context=".RecuperandoFotosActivity">

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical">

            <ImageView
                android:id="@+id/imageView1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content" />

            <TextView
                android:id="@+id/textView1"
                android:layout_width="match_parent"
```

```
            android:layout_height="wrap_content"
            android:gravity="left"
            android:textSize="30dp"
            android:textStyle="normal" />


        <ImageView
            android:id="@+id/imageView2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

        <TextView
            android:id="@+id/textView2"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="left"
            android:textSize="30dp"
            android:textStyle="normal" />


        <ImageView
            android:id="@+id/imageView3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

        <TextView
            android:id="@+id/textView3"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="left"
            android:textSize="30dp"
            android:textStyle="normal" />


        <ImageView
            android:id="@+id/imageView4"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

        <TextView
            android:id="@+id/textView4"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="left"
            android:textSize="30dp"
            android:textStyle="normal" />


        <ImageView
            android:id="@+id/imageView5"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

        <TextView
            android:id="@+id/textView5"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="left"
```

```
                android:textSize="30dp"
                android:textStyle="normal" />


            <ImageView
                android:id="@+id/imageView6"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content" />

            <TextView
                android:id="@+id/textView6"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="left"
                android:textSize="30dp"
                android:textStyle="normal" />

        </LinearLayout>

    </ScrollView>


</LinearLayout>
```

Código-fonte 6.5 – XML activity_recuperando_fotos
Fonte: Elaborado pelo autor (2020)

```
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import com.bumptech.glide.Glide
import kotlinx.android.synthetic.main.activity_recuperando_fotos.*

class RecuperandoFotosActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_recuperando_fotos)

        //-- Declarando um array com descritores das imagens
        var equipes =
arrayOf("Leão","Pinguim","Urso","Girafa","Tigre","Rato")

        //-- Adicionando o texto nos objetos visuais
        textView1.setText(equipes[0])
        textView2.setText(equipes[1])
        textView3.setText(equipes[2])
        textView4.setText(equipes[3])
        textView5.setText(equipes[4])
        textView6.setText(equipes[5])

        //-- Declarando as variáveis das imagens que serão
apresentados nos objetos visuais
        var i1 = "http://clipartbarn.com/wp-
content/uploads/2017/08/Clipart-animals-free-images.png"
        var i2 = "http://clipartbarn.com/wp-
content/uploads/2017/08/Animal-clipart-black-and-white-free-images-
2.png"
```

```
        var i3 = "http://clipartbarn.com/wp-
content/uploads/2017/08/Free-animal-clipart-for-teachers-animales-
predise-ados.png"

        var i4 = "http://clipartbarn.com/wp-
content/uploads/2017/08/Animal-clipart-images-on.jpg"
        var i5 = "http://clipartbarn.com/wp-
content/uploads/2017/08/Free-animal-clip-art-clipart-2.gif"
        var i6 = "http://clipartbarn.com/wp-
content/uploads/2017/08/Clip-art-animals-woodland-images-on.jpg"

        //-- Utilizando o framework Glide para recuperar a informação
        //-- das imagens que estão fora do dispositivo mobile
        Glide.with(this).load(i1).into(imageView1!!)
        Glide.with(this).load(i2).into(imageView2!!)
        Glide.with(this).load(i3).into(imageView3!!)

        Glide.with(this).load(i4).into(imageView4!!)
        Glide.with(this).load(i5).into(imageView5!!)
        Glide.with(this).load(i6).into(imageView6!!)
    }
```

Código-fonte 6.6 – Classe RecuperandoFotosActivity
Fonte: Elaborado pelo autor (2020)

6.2 Utilizando mapas: MapView

Para utilizarmos mapas nos aplicativos Android, foram desenvolvidos exemplos para que você possa experimentar as principais funcionalidades do MapView, componente que viabiliza o uso de mapas. Nesta sessão, serão apresentadas, por meio do MapView, as unidades da FIAP, bem como *pins* coloridos para cada local. Perceba que, ao tocar em cada pino, serão apresentados os dados da unidade em questão. Leia, estude, aprofunde-se! Coloque mapas em suas aplicações e deixe o app mais interativo. A Figura “Utilizando MapView” apresenta um exemplo do uso do MapView e a Figura “Informações personalizadas no marcador” mostra o recurso de visualizar detalhes da localização. Todas as informações são provenientes da API do Google Maps.

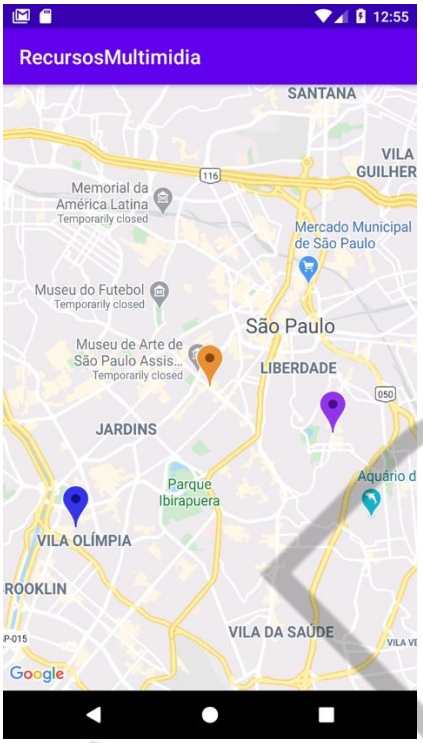

Figura 6.3 – Utilizando MapView
Fonte: Elaborado pelo autor (2020)

Figura 6.4 – Informações personalizadas no marcador
Fonte: Elaborado pelo autor (2018)

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
<!--
TODO: Before you run your application, you need a Google Maps API key.

To get one, follow this link, follow the directions and press "Create" at the
end:

https://console.developers.google.com/flows/enableapi?apiid=maps_android_backend&
keyType=CLIENT_SIDE_ANDROID&r=66:B1:A7:DE:C2:68:48:9B:28:66:9B:7F:AA:AE:6A:9E:21:
EE:AF:A2%3Bbr.com.fiap.recursosmultimdia

You can also add your credentials to an existing key, using these values:

Package name:
66:B1:A7:DE:C2:68:48:9B:28:66:9B:7F:AA:AE:6A:9E:21:EE:AF:A2

SHA-1 certificate fingerprint:
66:B1:A7:DE:C2:68:48:9B:28:66:9B:7F:AA:AE:6A:9E:21:EE:AF:A2

Alternatively, follow the directions here:
https://developers.google.com/maps/documentation/android/start#get-key

Once you have your key (it starts with "AIza"), replace the "google_maps_key"
```

```
string in this file.
-->
<string name="google_maps_key" templateMergeStrategy="preserve"
translatable="false">
    AIzaSyDVff6FC6jjSlJiS3xY-rBWJmOA7CWMhQU
</string>
</resources>
```

Código-fonte 6.7 – XML google_maps_api
Fonte: Elaborado pelo autor (2018)

Utilizando as ferramentas de refatoração do Android Studio, crie um arquivo de recurso chamado google_maps_api (File -> New -> Android Resource File) na pasta values, para que a chave de acesso da API possa ser declarada. Ela é necessária para o uso do MapView. Nesse arquivo criado, utilize o Código-fonte "XML google_maps_api". E, em seguida, crie uma nova activity chamada de utilizando_mapas_mapview e utilize o arquivo XML do Código-fonte "XML UtilizandoMapasActivity" e o trecho em Kotlin do Código-fonte "Classe UtilizandoMapasActivity".

É possível observar, nesse código, que o MapView é um componente que necessita de uma chamada assíncrona por meio de uma classe chamada OnMapReadyCallback. Essa chamada é invocada pelo método getMapAsync().

No momento em que o mapa é carregado, a call-back onMapReady é chamada e nela é feita a carga de todos os marcadores. Importante ressaltar que toda e qualquer chamada de componente da web necessita de operações assíncronas, partindo do pressuposto que a obtenção de dados garantidos da web é nativamente não-preemptiva.

```
<fragment
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".UtilizandoMapasActivity" />
```

Código-fonte 6.8 – XML activity_utilizando_mapas
Fonte: Elaborado pelo autor (2020)

```
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.graphics.Color
import android.graphics.Typeface
import android.view.Gravity
import android.view.View
```

```
import android.widget.LinearLayout
import android.widget.TextView
import com.google.android.gms.maps.CameraUpdateFactory
import com.google.android.gms.maps.GoogleMap
import com.google.android.gms.maps.OnMapReadyCallback
import com.google.android.gms.maps.SupportMapFragment
import com.google.android.gms.maps.model.BitmapDescriptorFactory
import com.google.android.gms.maps.model.LatLng
import com.google.android.gms.maps.model.Marker
import com.google.android.gms.maps.model.MarkerOptions

class UtilizandoMapasActivity : AppCompatActivity(),
OnMapReadyCallback {

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_utilizando_mapas)

        val mapFragment =
supportFragmentManager.findFragmentById(R.id.map) as
SupportMapFragment

        mapFragment.getMapAsync(this)
    }

    /**
     * Esta função manipula o mapa com todos os detalhes necessários
     * para a apresentação de informações, tais como, cores dos pins,
     * textos de títulos e complementos de endereços.
     *
     * Inserimos dados de latitute e lontitude fixos com as unidades
da FIAP
     */
    override fun onMapReady(googleMap: GoogleMap) {

        //-- vincula os objetos

        //-- prepara uma coleção de informações das unidades da FIAP
        //-- A informação \n irá pular uma linha
        val unidades = arrayOf(
            arrayOf(
                "FIAP Campus Vila Olimpia",
                "Rua Olimpíadas,186\nSão Paulo - SP\nCEP: 04551-000"
            ),
            arrayOf("FIAP Campus Paulista", "Av. Paulista,1106\nSão
Paulo - SP\nCEP: 01311-000"),
            arrayOf(
                "FIAP Campus Vila Mariana",
                "Av. Lins de Vasconcelos,1264\nSão Paulo - SP\nCEP:
01538-001"
            )
        )

        //--Adiciona a latitude e longitude da FIAP Campus Vila
Olimpia
        val fiapVilaOlimpia = LatLng(-23.5955843, -46.6851937)
```

```
        //--Adiciona a latitude e longitude da FIAP Campus Paulista
        val fiapPaulista = LatLng(-23.5643721, -46.652857)

        //--Adiciona a latitude e longitude da FIAP Campus Vila
Mariana
        val fiapVilaMariana = LatLng(-23.5746685, -46.6232043)

        //--Insere no objeto mapa os dados da unidade 1
        googleMap.addMarker(
            MarkerOptions()
                .position(fiapVilaOlimpia)
                .title(unidades[0][0])
                .snippet(unidades[0][1])

.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HU
E_BLUE))
        )

        //--Insere no objeto mapa os dados da unidade 2
        googleMap.addMarker(
            MarkerOptions()
                .position(fiapPaulista)
                .title(unidades[1][0])
                .snippet(unidades[1][1])

.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HU
E_ORANGE))
        )

        //--Insere no objeto mapa os dados da unidade 3
        googleMap.addMarker(
            MarkerOptions()
                .position(fiapVilaMariana)
                .title(unidades[2][0])
                .snippet(unidades[2][1])

.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HU
E_VIOLET))
        )

        //--Movimenta a camera para que a visualização aparece o mais
perto dos
        //--endereços das unidades. O valor float 12.5F indica a
distância da comera
        //--que pode varia entre 0.0F e 21.0F

googleMap.moveCamera(CameraUpdateFactory.newLatLngZoom(fiapPaulista,
12.5F));

        //--Configura a exibição dos títulos e endereços das unidades
FIAP
        //--de maneira personalizada
        googleMap.setInfoWindowAdapter(object :
GoogleMap.InfoWindowAdapter {

            override fun getInfoWindow(arg0: Marker): View? {
                return null
            }
```

```
            override fun getInfoContents(marker: Marker): View {

                val info = LinearLayout(applicationContext)
                info.orientation = LinearLayout.VERTICAL

                //--Título
                val title = TextView(applicationContext)
                title.setTextColor(Color.BLACK)
                title.gravity = Gravity.LEFT
                title.setTypeface(null, Typeface.BOLD)
                title.text = marker.title

                //--Complemento
                val snippet = TextView(applicationContext)
                snippet.setTextColor(Color.GRAY)
                snippet.text = marker.snippet

                //--Adiciona o titulo e o complemento na marca
                info.addView(title)
                info.addView(snippet)

                return info
            }
        })
    }
}
```

Código-fonte 6.9 – Classe UtilizandoMapasActivity
Fonte: Elaborado pelo autor (2020)

6.3 Localização do usuário com Location API

Nesta seção, é dada continuidade ao uso dos mapas, localizando coordenadas por meio dos pontos de latitude e longitude, marcando no mapa com um ícone personalizado. A rotina foi amplificada para que, a cada tempo, uma nova leitura de coordenadas aconteça e uma nova pinagem seja apresentada. Por exemplo, faça com que os aplicativos mostrem locais, caminhos e rotas. Essa é uma excelente ideia para aplicativos de relógios inteligentes, pois podemos marcar os pontos em que o usuário esteve. A Figura “Localização do usuário” apresenta um exemplo da localização de usuário.

Crie uma nova Activity com o nome localizacao_do_usuario_com_location_api e utilize como base o trecho XML do Código-fonte “XML activity_localizacao_usuario” e trecho em Kotlin do Código-fonte “Classe LocalizacaoUsuarioActivity”. Observe o uso do LocationManager, o qual fornece os subsídios para a marcação, no mapa, da

posição atual do dispositivo. Importante ressaltar, para os casos no teste em simulador, que a localização esteja ativa.

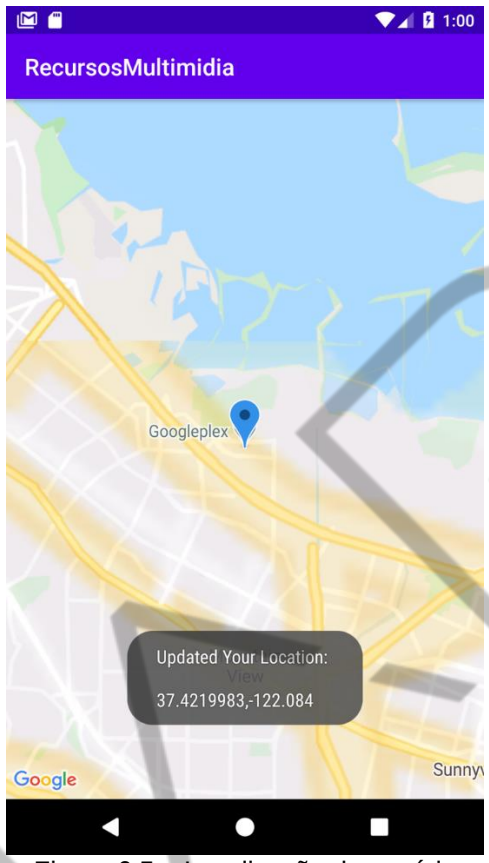

Figura 6.5 – Localização do usuário
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<fragment
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".LocalizacaoUsuarioActivity" />
```

Código-fonte 6.10 – XML activity_localizacao_usuario
Fonte: Elaborado pelo autor (2020)

```
import android.Manifest
import androidx.appcompat.app.AppCompatActivity
import android.annotation.SuppressLint
import android.content.Context
import android.content.pm.PackageManager
import android.graphics.Color
import android.graphics.Typeface
import android.location.Location
import android.location.LocationManager
```

```
import android.os.Bundle
import android.view.Gravity
import android.view.View
import android.widget.LinearLayout
import android.widget.TextView
import android.widget.Toast
import androidx.core.app.ActivityCompat
import com.google.android.gms.common.ConnectionResult
import com.google.android.gms.common.api.GoogleApiClient
import com.google.android.gms.location.FusedLocationProviderClient
import com.google.android.gms.location.LocationRequest
import com.google.android.gms.location.LocationServices
import com.google.android.gms.maps.CameraUpdateFactory
import com.google.android.gms.maps.GoogleMap
import com.google.android.gms.maps.OnMapReadyCallback
import com.google.android.gms.maps.SupportMapFragment
import com.google.android.gms.maps.model.BitmapDescriptorFactory
import com.google.android.gms.maps.model.LatLng
import com.google.android.gms.maps.model.Marker
import com.google.android.gms.maps.model.MarkerOptions

class LocalizacaoUsuarioActivity : AppCompatActivity(),
    GoogleApiClient.ConnectionCallbacks,
GoogleApiClient.OnConnectionFailedListener,
    com.google.android.gms.location.LocationListener,
OnMapReadyCallback {
    private lateinit var mapa: GoogleMap
    private var mGoogleApiClient: GoogleApiClient? = null
    private var mLocation: Location? = null
    private var mLocationManager: LocationManager? = null
    private var mLocationRequest: LocationRequest? = null
    private var locationManager: LocationManager? = null
    private var fusedLocationClient: FusedLocationProviderClient? =
null
    val isLocationEnabled: Boolean
        get() {
            locationManager =
                getSystemService(Context.LOCATION_SERVICE) as
LocationManager
            return locationManager!!.isProviderEnabled(
                LocationManager.GPS_PROVIDER
            ) ||
locationManager!!.isProviderEnabled(LocationManager.NETWORK_PROVIDER)
        }

    val UPDATE_INTERVAL = (2 * 10000).toLong()
    val FASTEST_INTERVAL: Long = 2000

    //-----------------------------------------------------------
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_localizacao_usuario)
        val mapFragment =
supportFragmentManager.findFragmentById(R.id.map) as
SupportMapFragment
        mapFragment.getMapAsync(this)
        mGoogleApiClient =
GoogleApiClient.Builder(this).addConnectionCallbacks(this)
```

```
.addOnConnectionFailedListener(this).addApi(LocationServices.API)
            .build()
        fusedLocationClient =
LocationServices.getFusedLocationProviderClient(this)
        mLocationManager =
this.getSystemService(Context.LOCATION_SERVICE) as LocationManager
        checkLocation()
        startLocationUpdates()
    }    //----------------------------------------------------------

    @SuppressLint("MissingPermission")
    override fun onConnected(p0: Bundle?) {

        if (ActivityCompat.checkSelfPermission(
                this,
                Manifest.permission.ACCESS_FINE_LOCATION
            ) != PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(
                this,
                Manifest.permission.ACCESS_COARSE_LOCATION
            ) != PackageManager.PERMISSION_GRANTED
        ) {
            if (ActivityCompat.shouldShowRequestPermissionRationale(
                    this,
                    Manifest.permission.ACCESS_FINE_LOCATION
                )
            ) {
                ActivityCompat.requestPermissions(
                    this,
                    arrayOf(Manifest.permission.ACCESS_FINE_LOCATION),
1
                )
            } else {
                ActivityCompat.requestPermissions(
                    this,
                    arrayOf(Manifest.permission.ACCESS_FINE_LOCATION),
1
                )
            }
        }
    }    //----------------------------------------------------------

    override fun onConnectionSuspended(i: Int) {
        mGoogleApiClient!!.connect()
    }    //----------------------------------------------------------

    override fun onConnectionFailed(connectionResult:
ConnectionResult) {}

    //----------------------------------------------------------
    override fun onStart() {
        super.onStart()
        if (mGoogleApiClient != null) {
            mGoogleApiClient!!.connect()
        }
    }    //----------------------------------------------------------
```

```
    override fun onStop() {
        super.onStop()
        if (mGoogleApiClient!!.isConnected()) {
            mGoogleApiClient!!.disconnect()
        }
    }    //------------------------------------------------------------

    @SuppressLint("MissingPermission")
    protected fun startLocationUpdates() {

        mLocationRequest =

LocationRequest.create().setPriority(LocationRequest.PRIORITY_HIGH_ACC
URACY)

.setInterval(UPDATE_INTERVAL).setFastestInterval(FASTEST_INTERVAL)
        if (ActivityCompat.checkSelfPermission(
                this,
                Manifest.permission.ACCESS_FINE_LOCATION
            ) != PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(
                this,
                Manifest.permission.ACCESS_COARSE_LOCATION
            ) != PackageManager.PERMISSION_GRANTED
        ) {
            return
        }
        fusedLocationClient?.lastLocation?.addOnSuccessListener {
location: Location? ->
            location?.let {
                val msg =
                    "Updated Your Location:\n\n" +
java.lang.Double.toString(location.latitude) + "," +
java.lang.Double.toString(
                        location.longitude
                    )
                Toast.makeText(this, msg, Toast.LENGTH_SHORT).show()
                val latLng = LatLng(location.latitude,
location.longitude)
                updateMap(mapa, latLng)
            } ?: Toast.makeText(this, "Location not Detected",
Toast.LENGTH_SHORT).show()

        }

    }    //------------------------------------------------------------

    override fun onLocationChanged(location: Location) {

    }    //------------------------------------------------------------

    private fun checkLocation(): Boolean {
        return isLocationEnabled
    }//------------------------------------------------------------

    override fun onMapReady(googleMap: GoogleMap) {
```

```
        updateMap(googleMap, null)
    }    //-------------------------------------------------------------

    fun updateMap(googleMap: GoogleMap, latLng: LatLng?) {

//-- vincula os objetos
        mapa = googleMap
//--Adiciona a latitude e longitude da FIAP Campus Vila Mariana
        var myTitle = "FIAP Campus Vila Mariana"
        var mySnippet = "Av. Lins de Vasconcelos,1264\nSão Paulo -
SP\nCEP: 01538-001"
        var myLocation = LatLng(-23.5746685, -46.6232043)
        if (latLng != null) {
            myLocation = latLng
            myTitle = "Localização capturada"
            mySnippet = "Veja os detalhes\nda sua localização."
        }
//-- Faixa de cores dos Pins no Mapa
        var bitmap =
            arrayOf(0.0F, 30.0F, 60.0F, 120.0F, 180.0F, 210.0F,
240.0F, 270.0F, 300.0F, 330.0F)
        var bitmapSorted = bitmap[((Math.random() *
bitmap.size).toInt())]
//--Insere no objeto mapa
        mapa.addMarker(

MarkerOptions().position(myLocation).title(myTitle).snippet(mySnippet)

.icon(BitmapDescriptorFactory.defaultMarker(bitmapSorted))
        )
//--Movimenta a camera para que a visualização aparece o mais perto
dos        //--endereços das unidades. O valor float 12.5F indica a
distância da comera        //--que pode varia entre 0.0F e 21.0F
        mapa.moveCamera(
            CameraUpdateFactory.newLatLngZoom(
                myLocation,
                12.5F
            )
        );
        //--Configura a exibição dos títulos e endereços das unidades
FIAP        //--de maneira personalizada
        mapa.setInfoWindowAdapter(object : GoogleMap.InfoWindowAdapter
{
            override fun getInfoWindow(arg0: Marker): View? {
                return null
            }

            override fun getInfoContents(marker: Marker): View {
                val info = LinearLayout(applicationContext)
                info.orientation = LinearLayout.VERTICAL
//--Título
                val title = TextView(applicationContext)
                title.setTextColor(Color.BLACK)
                title.gravity = Gravity.LEFT
                title.setTypeface(null, Typeface.BOLD)
                title.text = marker.title

//--Complemento
```

```
                val snippet = TextView(applicationContext)
                snippet.setTextColor(Color.GRAY)
                snippet.text = marker.snippet
//--Adiciona o titulo e o complemento na marca
                info.addView(title)
                info.addView(snippet)
                return info
            }
        })
    }
}
//-------------------------------------------------------------
```

Código-fonte 6.11 – Classe LocalizacaoUsuarioActivity
Fonte: Elaborado pelo autor (2020)

6.4 Rotas e Pontos de Interesse

Gerenciar rotas é uma atividade diária na vida das pessoas, assim como descobrir de que maneira é possível chegar a um compromisso pessoal o mais rápido possível. Essas atividades nos ajudam a ganhar tempo. Marcar um ponto turístico em um mapa é outro fato corriqueiro, ou seja, rotas e pontos são importantes situações que poderemos programar em nossos aplicativos.

Nesta seção, você descobrirá como marcar dois pontos e estabelecer uma rota entre eles, desenhando no mapa por onde ir e por quais ruas passar. Utilizamos as unidades da FIAP para desenhar as rotas. Verifique as Activities que necessitam ser criadas e observe o funcionamento desse recurso oferecido pela API do Google Maps.

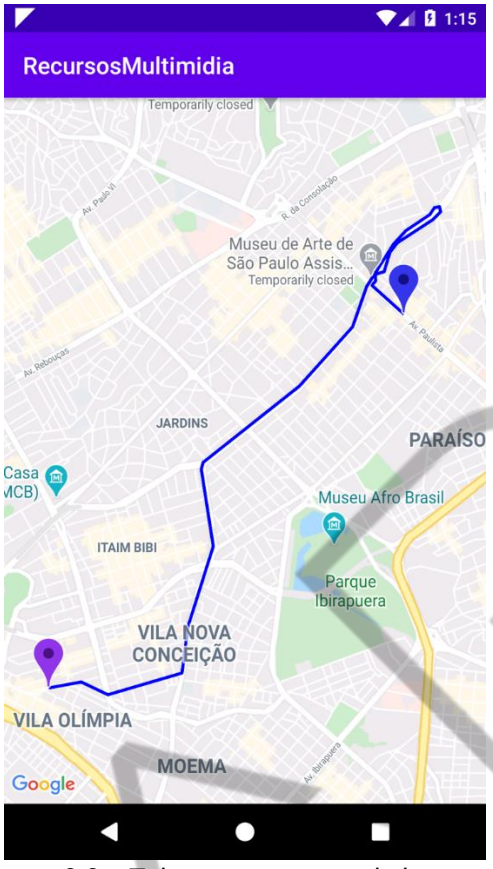

Figura 6.6 – Tela rotas e pontos de interesse
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<fragment
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".PrimeiraRotaActivity" />
```

Código-fonte 6.12 – XML activity_primeira_rota
Fonte: Elaborado pelo autor (2020)

```
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.graphics.Color
import android.graphics.Typeface
import android.view.Gravity
import android.view.View
import android.widget.LinearLayout
import android.widget.TextView
import com.beust.klaxon.*
import com.google.android.gms.maps.CameraUpdateFactory
import com.google.android.gms.maps.GoogleMap
import com.google.android.gms.maps.OnMapReadyCallback
import com.google.android.gms.maps.SupportMapFragment
import com.google.android.gms.maps.model.*
import org.jetbrains.anko.async
import org.jetbrains.anko.uiThread
import java.net.URL

class PrimeiraRotaActivity : AppCompatActivity(), OnMapReadyCallback {

    /**
     * Nesta rotina iremos instanciar um Mapa com 2 pontos e desenhar
uma rota entre os
     * o ponto1 e ponto2 consultando as coordenadas via Google Maps
     */

        //-- Objeto contendo o Mapa do Google
        private var mMap: GoogleMap? = null

        //-- Iniciando a rotina
        override fun onCreate(savedInstanceState: Bundle?) {

            super.onCreate(savedInstanceState)
            setContentView(R.layout.activity_primeira_rota)

            val mapFragment = supportFragmentManager
                .findFragmentById(R.id.map) as SupportMapFragment

            mapFragment.getMapAsync(this)
        }

        /**
         * método principal do Google Maps
         */
        override fun onMapReady(googleMap: GoogleMap) {

            //-- Atualiza o objeto do Mapa
            mMap = googleMap

            //-- Prepara um objeto com latitude e longitude
            val LatLongB = LatLngBounds.Builder()

            //-- prepara uma coleção de informações das unidades da
FIAP
            //-- A informação \n irá pular uma linha
            val unidades = arrayOf(
                arrayOf(
                    "FIAP Campus Vila Olimpia",
```

```
                    "Rua Olimpíadas,186\nSão Paulo - SP\nCEP: 04551-
000"
                ),
                arrayOf(
                    "FIAP Campus Paulista",
                    "Av. Paulista,1106\nSão Paulo - SP\nCEP: 01311-
000"
                ),
                arrayOf(
                    "FIAP Campus Vila Mariana",
                    "Av. Lins de Vasconcelos,1264\nSão Paulo -
SP\nCEP: 01538-001"
                )
            )

            //--Adiciona a latitude e longitude da FIAP Campus Vila
Mariana
            val fiap_campus_vila_mariana = LatLng(-23.5746685, -
46.6232043)

            //--Adiciona a latitude e longitude da FIAP Campus Vila
Olimpia
            val fiap_campus_vila_olimpia = LatLng(-23.5955843, -
46.6851937)

            //--Adiciona a latitude e longitude da FIAP Campus
Paulista
            val fiap_campus_paulista = LatLng(-23.5643721, -46.652857)

            //-- Selecionando informações
            val pontoA = fiap_campus_paulista;
            val pontoB = fiap_campus_vila_olimpia;

            val unidadePontoA = unidades[1]
            val unidadePontoB = unidades[0]

            //-- Unidade Ponto A
            mMap!!.addMarker(
                MarkerOptions()
                    .position(pontoA)
                    .title(unidadePontoA[0])
                    .snippet(unidadePontoA[1])

.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HU
E_BLUE))
            )

            //-- Unidade Ponto B
            mMap!!.addMarker(
                MarkerOptions()
                    .position(pontoB)
                    .title(unidadePontoB[0])
                    .snippet(unidadePontoB[1])

.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HU
E_VIOLET))
            )
```

```
            //-- Programando a rota entre o Ponto A e Ponto B
            val url = getURL(pontoA, pontoB)

            //-- Processando as informações visual da cor da linha e
da largura da linha
            val options = PolylineOptions()
            options.color(Color.BLUE)
            options.width(7f)

            //-- Sincronizando pedido das informações com o Google
Mapa via INTERNET
            async {

                val result = URL(url).readText()

                uiThread {

                    val parser: Parser = Parser()
                    val stringBuilder: StringBuilder =
StringBuilder(result)
                    val json: JsonObject = parser.parse(stringBuilder)
as JsonObject
                    val routes = json.array<JsonObject>("routes")
                    if ((routes?.size ?: 0) <= 0) {
                        return@uiThread
                    }

                    val points = routes!!["legs"]["steps"][0] as
JsonArray<JsonObject>
                    val polypts =
                        points.flatMap {
decodePoly(it.obj("polyline")?.string("points")!!) }

                    //-- Processando as informações do Ponto A
                    options.add(pontoA)
                    LatLongB.include(pontoA)
                    for (point in polypts) {
                        options.add(point)
                        LatLongB.include(point)
                    }

                    //-- Processando as informações do Ponto B
                    options.add(pontoB)
                    LatLongB.include(pontoB)
                    val bounds = LatLongB.build()

                    //-- Adicionando a rota no mapa
                    mMap!!.addPolyline(options)

                    //-- Centralizando o Mapa

mMap!!.moveCamera(CameraUpdateFactory.newLatLngBounds(bounds, 100))

                    //--Configura a exibição dos títulos e endereços
das unidades FIAP
                    //--de maneira personalizada
                    mMap!!.setInfoWindowAdapter(object :
GoogleMap.InfoWindowAdapter {
```

```
                        override fun getInfoWindow(arg0: Marker):
View? {
                            return null
                        }

                        override fun getInfoContents(marker: Marker):
View {

                            val info =
LinearLayout(applicationContext)
                            info.orientation = LinearLayout.VERTICAL

                            //--Título
                            val title = TextView(applicationContext)
                            title.setTextColor(Color.BLACK)
                            title.gravity = Gravity.LEFT
                            title.setTypeface(null, Typeface.BOLD)
                            title.text = marker.title

                            //--Complemento
                            val snippet = TextView(applicationContext)
                            snippet.setTextColor(Color.GRAY)
                            snippet.text = marker.snippet

                            //--Adiciona o titulo e o complemento na
marca
                            info.addView(title)
                            info.addView(snippet)

                            return info
                        }
                    })

                }
            }
        }

        //-- Coletando os dados do PontoA e PontoB via URL
        private fun getURL(from: LatLng, to: LatLng): String {

            val origin = "origin=" + from.latitude + "," +
from.longitude
            val dest = "destination=" + to.latitude + "," +
to.longitude
            val sensor = "sensor=false"
            var key: String? = "null"
            key?.let {
                val params = "$origin&$dest&$sensor&key=$it"
                return
"https://maps.googleapis.com/maps/api/directions/json?$params"
            }
            throw IllegalArgumentException("TROQUE O VALOR DE KEY")
        }

        //-- Decodificando os pontos
        private fun decodePoly(encoded: String): List<LatLng> {
            val poly = ArrayList<LatLng>()
```

```
            var index = 0
            val len = encoded.length
            var lat = 0
            var lng = 0

            while (index < len) {
                var b: Int
                var shift = 0
                var result = 0
                do {
                    b = encoded[index++].toInt() - 63
                    result = result or (b and 0x1f shl shift)
                    shift += 5
                } while (b >= 0x20)
                val dlat = if (result and 1 != 0) (result shr 1).inv()
else result shr 1
                lat += dlat

                shift = 0
                result = 0
                do {
                    b = encoded[index++].toInt() - 63
                    result = result or (b and 0x1f shl shift)
                    shift += 5
                } while (b >= 0x20)
                val dlng = if (result and 1 != 0) (result shr 1).inv()
else result shr 1
                lng += dlng

                val p = LatLng(
                    lat.toDouble() / 1E5,
                    lng.toDouble() / 1E5
                )
                poly.add(p)
            }

            return poly
        }
        //-----------------------------------------------------Fim da
rotina
    }
```

Código-fonte 6.13 – Classe PrimeiraRotasActivity
Fonte: Elaborado pelo autor (2020)

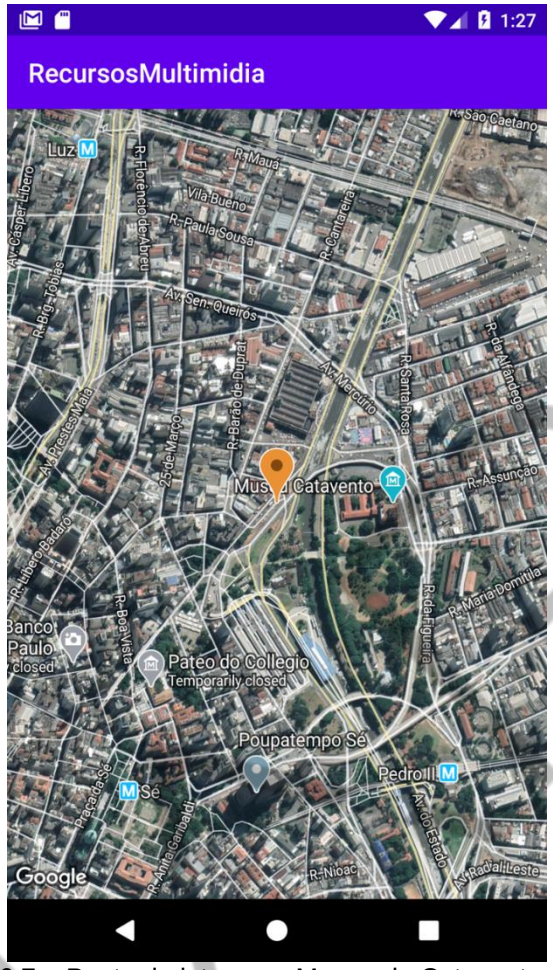

Figura 6.7 – Ponto de interesse Museu do Catavento Cultural
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<fragment

xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"

android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Utilizando_Mapas_MapView" /
```

Código-fonte 6.14 – XML activity_pontos_interesse
Fonte: Elaborado pelo autor (2020)

```
import android.graphics.Color
import android.graphics.Typeface
import android.os.Bundle
import android.support.v7.app.AppCompatActivity
import android.view.Gravity
import android.view.View
import android.widget.LinearLayout
```

```
import android.widget.TextView
import com.google.android.gms.maps.CameraUpdateFactory
import com.google.android.gms.maps.GoogleMap
import com.google.android.gms.maps.OnMapReadyCallback
import com.google.android.gms.maps.SupportMapFragment
import
com.google.android.gms.maps.model.BitmapDescriptorFactory
import com.google.android.gms.maps.model.LatLng
import com.google.android.gms.maps.model.Marker
import com.google.android.gms.maps.model.MarkerOptions

class Rotas_e_Pontos_de_Interesse_Ponto1 :
AppCompatActivity(), OnMapReadyCallback {

    private lateinit var mapa: GoogleMap

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)

setContentView(R.layout.rotas_e_pontos_de_interesse_ponto1)

        val mapFragment =
        supportFragmentManager.findFragmentById(R.id.map) as
SupportMapFragment

        mapFragment.getMapAsync(this)
    }

    /**
     * Função para o gerenciamento de informações no mapa
     */
    override fun onMapReady(googleMap: GoogleMap) {

        //-- vincula os objetos
        mapa = googleMap

        //-- Dados do ponto de interesse
        val ponto_de_interesse_informacoes =
        arrayOf("Catavento Cultural",
                "Parque Dom Pedro II\nAv. Mercúrio, s/n\nSão
Paulo - SP")

        //--Coordenadas do Ponto de Interesse
        val ponto_de_interesse_latitude_longitude = LatLng(-
23.5440055,-46.629888)

        //--Insere no objeto mapa os dados do ponto do
interesse
        mapa.addMarker(MarkerOptions()
```

```
.position(ponto_de_interesse_latitude_longitude)
                .title(ponto_de_interesse_informacoes[0])
                .snippet(ponto_de_interesse_informacoes[1])

.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorF
actory.HUE_ORANGE))
        )

        //--Movimenta a camera para que a visualização
aparece o mais perto dos
        //--endereços das unidades. O valor float 12.5F
indica a distância da comera
        //--que pode varia entre 0.0F e 21.0F

mapa.moveCamera(CameraUpdateFactory.newLatLngZoom(ponto_de_in
teresse_latitude_longitude, 15.5F));

        //-- MAP_TYPE_NORMAL
        //-- MAP_TYPE_TERRAIN
        //-- MAP_TYPE_HYBRID
        //-- MAP_TYPE_NONE
        //-- MAP_TYPE_SATELLITE
        mapa.setMapType(GoogleMap.MAP_TYPE_HYBRID);

        //--Configura a exibição de informações de maneira
personalizada
        mapa.setInfoWindowAdapter(object :
GoogleMap.InfoWindowAdapter {

            override fun getInfoWindow(arg0: Marker): View? {
                return null
            }

            override fun getInfoContents(marker: Marker):
View {

                val info = LinearLayout(applicationContext)
                info.orientation = LinearLayout.VERTICAL

                //--Título
                val title = TextView(applicationContext)
                title.setTextColor(Color.BLACK)
                title.gravity = Gravity.LEFT
                title.setTypeface(null, Typeface.BOLD)
                title.text = marker.title

                //--Complemento
                val snippet = TextView(applicationContext)
                snippet.setTextColor(Color.GRAY)
                snippet.text = marker.snippet
```

```
                //--Adiciona o titulo e o complemento na
marca
                info.addView(title)
                info.addView(snippet)

                return info
            }
        })
    }
}
```

Código-fonte 6.15 – Classe PontosInteresseActivity
Fonte: Elaborado pelo autor (2020)

6.5 Usando WebView em seus Apps

Navegar pela web também é um recurso muito utilizado em aplicativos para dispositivos móveis. Nesta seção do capítulo, é apresentado o componente gráfico WebView, que possibilita a implementação de um navegador como uma view do Android. Por meio da WebView é possível efetuar adaptações de funcionalidades da web em aplicativos desenvolvidos para o sistema operacional Android. O Código-fonte “XML activity_usando_web_view” e o Código-fonte em Kotlin “Classe UsandoWebView” apresentam um exemplo de como invocar a WebView.

Figura 6.8 – Utilizando WebView
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginLeft="0dp"
    android:layout_marginRight="0dp"
    android:orientation="vertical"
    tools:context=".UsandoWebViewActivity">

    <WebView
        android:id="@+id/webview"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:ignore="MissingConstraints"
        tools:layout_editor_absoluteX="8dp"
        tools:layout_editor_absoluteY="8dp" />

</LinearLayout>
```

Código-fonte 6.16 – XML activity_usando_web_view
Fonte: Elaborado pelo autor (2020)

```
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.webkit.WebView
import android.webkit.WebViewClient
import kotlinx.android.synthetic.main.activity_usando_web_view.*

class UsandoWebViewActivity : AppCompatActivity() {

        override fun onCreate(savedInstanceState: Bundle?) {

            super.onCreate(savedInstanceState)
            setContentView(R.layout.activity_usando_web_view)

            webview!!.webViewClient = object : WebViewClient() {
                override fun shouldOverrideUrlLoading(view: WebView?,
url: String?): Boolean {
                    view?.loadUrl(url)
                    return true
                }
            }

            webview!!.getSettings().setJavaScriptEnabled(true)
            webview!!.loadUrl("https://www.fiap.com.br/")
        }
    }
```

Código-fonte 6.17 – Classe UsandoWebViewActivity
Fonte: Elaborado pelo autor (2020)

6.6 Sons: AudioManager

Como apresentar sons personalizados em nossos aplicativos? Nesta seção, apresentaremos a pasta de recursos chamado "raw", além das rotinas de programação em Kotlin necessárias para a execução desses sons que desejamos em nossos aplicativos mobile no Android. Prepare-se para desenvolver instrumentos musicais, jogos lúdicos para o público infantil, despertadores digitais, avisos para tomar medicamentos, entre outras ideias.

A execução de áudio é feita por meio do componente MediaPlayer, no qual o áudio utilizado e as imagens dos ícones devem estar importados no projeto da solução Android.

Imagens na pasta drawable e sons na pasta raw

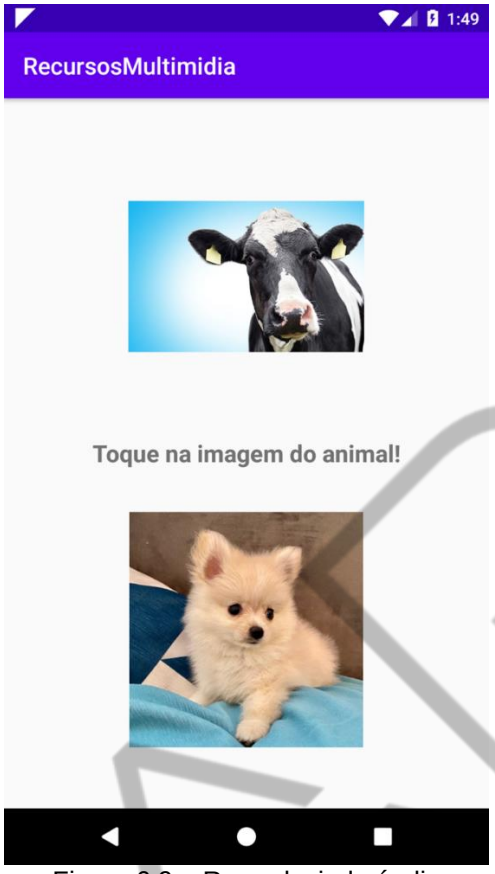

Figura 6.9 – Reproduzindo áudio
Fonte: Elaborado pelo autor (2020)

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical"
    android:gravity="center"
    android:padding="10dp"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:onClick="soundCow"
        android:layout_width="200dp"
        android:layout_height="200dp"
        app:srcCompat="@drawable/cow" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="100dp"
        android:layout_gravity="center"
        android:gravity="center_vertical|center_horizontal"
        android:text="Toque na imagem do animal!"
        android:textSize="20dp"
        android:textStyle="bold" />

    <ImageView
        android:onClick="soundDog"
        android:layout_width="200dp"
```

```
        android:layout_height="200dp"
        app:srcCompat="@drawable/dog" />

</LinearLayout>
```

Código-fonte 6.18 – XML activity_sons_audio
Fonte: Elaborado pelo autor (2020)

```
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.media.MediaPlayer
import android.view.View

class SonsAudioActivity : AppCompatActivity() {

    var cow: MediaPlayer? = null
    var dog: MediaPlayer? = null

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_sons_audio)

        cow = MediaPlayer.create(this, R.raw.cow)
        dog = MediaPlayer.create(this, R.raw.dog)
    }

    fun soundCow(view: View) {
        //if(dog != null) {
        //    dog!!.stop()
        //}
        cow!!.start()
    }

    fun soundDog(view: View) {
        //if(cow != null) {
        //    cow!!.stop()
        //}
        dog!!.start()
    }
}
```

Código-fonte 6.19 – Classe SonsAudioActivity
Fonte: Elaborado pelo autor (2020)

6.7 Vídeos: MediaPlayer

Na seção anterior, falamos sobre sons, e, nesta, abordaremos o uso de vídeos. Estamos vivendo uma geração ainda mais criativa que a anterior, a forma de consumir informações atualmente é feita por meio de vídeo, Youtube, Instagram TV, SnapChat, entre outros. Nesse caso, o componente utilizado é o VideoView, o qual permite a

execução de vídeos externos disponíveis na Web. Verifique os *listeners* que são implementados para a execução do vídeo nas suas etapas da execução (entende-se que o consumo de *streaming* seja uma ação assíncrona).

Para que o conhecimento seja ainda mais ampliado, nas rotinas seguintes você utilizará vídeos que estão na internet e poderá colocá-los em seus aplicativos Android:

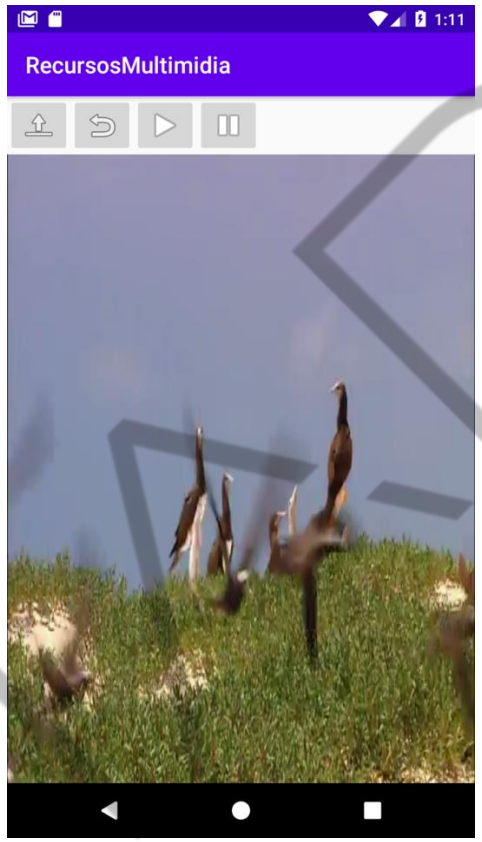

Figura 6.10 – Executando vídeo
Fonte: Elaborado pelo autor (2020)

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <ImageButton
            android:id="@+id/btnonce"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
```

```
            app:srcCompat="@android:drawable/ic_menu_upload_you_tube"
/>

        <ImageButton
            android:id="@+id/btnconti"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            app:srcCompat="@android:drawable/ic_menu_revert" />

        <ImageButton
            android:id="@+id/btnplay"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            app:srcCompat="@android:drawable/ic_media_play" />

        <ImageButton
            android:id="@+id/btnstop"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            app:srcCompat="@android:drawable/ic_media_pause" />



    </LinearLayout>

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <VideoView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_alignParentBottom="true"
            android:layout_alignParentEnd="true"
            android:layout_alignParentLeft="true"
            android:layout_alignParentRight="true"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:id="@+id/vv"/>

        <ProgressBar
            android:layout_width="200dp"
            android:layout_height="200dp"
            android:id="@+id/progrss"
            android:visibility="gone"
            android:layout_centerInParent="true"/>

    </RelativeLayout>

</LinearLayout>
```

Código-fonte 6.20 – XML activity_videos_media_player
Fonte: Elaborado pelo autor (2020)

```
import android.net.Uri
import android.os.Bundle
import android.view.View
import android.widget.MediaController
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_videos_media_player.*

class VideosMediaPlayerActivity : AppCompatActivity() {

    private var uri: Uri? = null

    private var isContinuously = false

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_videos_media_player)

        val mediaController = MediaController(this)
        mediaController.setAnchorView(vv)

        val uriPath =
"https://archive.org/download/WildlifeSampleVideo/Wildlife.mp4"

        uri = Uri.parse(uriPath)

        vv!!.setOnCompletionListener {
            if (isContinuously) {
                vv!!.start()
            }
        }

        btnstop!!.setOnClickListener { vv!!.pause() }
        btnplay!!.setOnClickListener { vv!!.start() }

        btnonce!!.setOnClickListener {
            isContinuously = false
            progrss!!.visibility = View.VISIBLE
            vv!!.setMediaController(mediaController)
            vv!!.setVideoURI(uri)
            vv!!.requestFocus()
            vv!!.start()
        }

        btnconti!!.setOnClickListener {
            isContinuously = true
            progrss!!.visibility = View.VISIBLE
            vv!!.setMediaController(mediaController)
            vv!!.setVideoURI(uri)
            vv!!.requestFocus()
            vv!!.start()
        }

        vv!!.setOnPreparedListener { progrss!!.visibility = View.GONE
}
```

```
    }
```

Código-fonte 6.21 – Classe VideosMediaPlayerActivity
Fonte: Elaborado pelo autor (2020)

6.8 Animação em views com View Animation

Além de todos os recursos de multimídia até o momento apresentados, temos, ainda, uma lista com dez rotinas de programação em XML para executar animações em objetos e telas para os aplicativos mobile Android.

Estude como as rotinas de programação foram desenvolvidas e aplique esses detalhes em suas aplicações, chame a atenção do seu usuário, não somente com mensagens de aviso, mas como uma elaborada animação. Crie uma nova pasta de recursos chamada "anim" dentro de da pasta *res* e estude cada uma das animações abaixo desenvolvidas em XML e depois aplique na rotina de programação em Kotlin.

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:shareInterpolator="false" >
    <alpha android:duration="500"
        android:fromAlpha="0.0"
        android:toAlpha="1.0" />
</set>
```

Código-fonte 6.22 – Animação Fade in
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:fillAfter="true" >
    <alpha android:duration="1000"
        android:fromAlpha="1.0"
        android:interpolator="@android:anim/accelerate_interpolator"
        android:toAlpha="0.0" />
</set>
```

Código-fonte 6.23 – Animação Fade out
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <alpha android:fromAlpha="0.0"
        android:toAlpha="1.0"
        android:interpolator="@android:anim/accelerate_interpolator"
        android:duration="600"
        android:repeatMode="reverse"
        android:repeatCount="infinite"/>
</set>
```

Código-fonte 6.24 – Animação blink
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:interpolator="@android:anim/linear_interpolator"
    android:fillAfter="true">
    <translate android:fromXDelta="0%p"
        android:toXDelta="75%p"
        android:duration="800" />
</set>
```

Código-fonte 6.25 – Animação move
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:fillAfter="true"
    android:interpolator="@android:anim/bounce_interpolator">
    <scale android:duration="500"
        android:fromXScale="1.0"
        android:fromYScale="0.0"
        android:toXScale="1.0"
        android:toYScale="1.0" />
</set>
```

Código-fonte 6.26 – Animação bounce
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <rotate android:fromDegrees="0"
        android:toDegrees="360"
        android:pivotX="50%"
        android:pivotY="50%"
        android:duration="600"
        android:repeatMode="restart"
        android:repeatCount="infinite"
        android:interpolator="@android:anim/cycle_interpolator"/>

</set>
```

Código-fonte 6.27 – Animação Rotate
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:fillAfter="true" >
    <scale android:duration="500"
        android:fromXScale="1.0"
        android:fromYScale="1.0"
        android:interpolator="@android:anim/linear_interpolator"
        android:toXScale="1.0"
        android:toYScale="0.0" />

</set>
```

Código-fonte 6.28 – Animação Slide up
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:fillAfter="true">
    <scale android:duration="500"
        android:fromXScale="1.0"
        android:fromYScale="0.0"
        android:interpolator="@android:anim/linear_interpolator"
        android:toXScale="1.0"
        android:toYScale="1.0" />

</set>
```

Código-fonte 6.29 – Animação Slide down
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:fillAfter="true" >
    <scale xmlns:android="http://schemas.android.com/apk/res/android"
        android:duration="1000"
        android:fromXScale="1"
        android:fromYScale="1"
        android:pivotX="50%"
        android:pivotY="50%"
        android:toXScale="3"
        android:toYScale="3" />
</set>
```

Código-fonte 6.30 – Animação Zoom in
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:fillAfter="true" >
    <scale xmlns:android="http://schemas.android.com/apk/res/android"
        android:duration="1000"
        android:fromXScale="1.0"
        android:fromYScale="1.0"
        android:pivotX="50%"
        android:pivotY="50%"
        android:toXScale="0.5"
        android:toYScale="0.5" />
</set>
```

Código-fonte 6.31 – Animação Zoom out
Fonte: Elaborado pelo autor (2020)

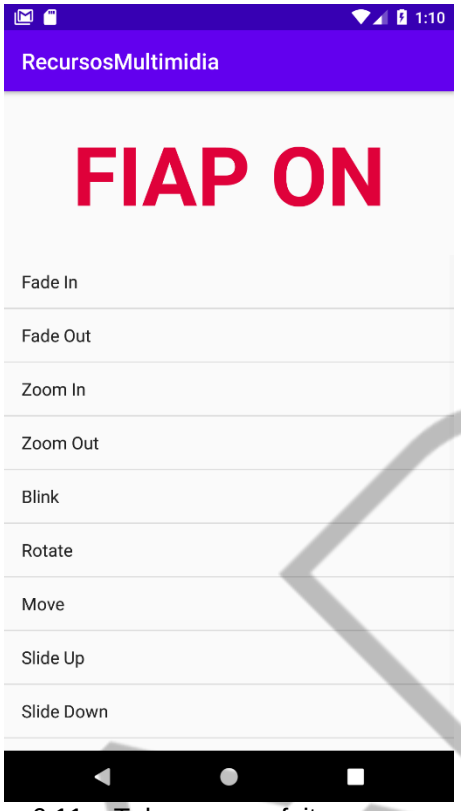

Figura 6.11 – Tela com os efeitos programados
Fonte: Elaborado pelo autor (2020)

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".AnimacaoActivity">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="150dp"
        android:id="@+id/tv"
        android:textAlignment="center"
        android:textSize="75dp"
        android:text="FIAP ON"
        android:textColor="#DF013A"
        android:textStyle="bold"
        android:paddingTop="25dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true" />

    <ListView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/lv"
        android:layout_below="@+id/tv"/>

</RelativeLayout>
```

Código-fonte 6.32 – XML activity_animacao
Fonte: Elaborado pelo autor (2020)

```
package com.example.android.recursosmultimidia

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.animation.AnimationUtils
import android.widget.AdapterView
import android.widget.ArrayAdapter
import android.widget.ListView
import android.widget.TextView
import kotlinx.android.synthetic.main.activity_animacao.*

class AnimacaoActivity : AppCompatActivity() {

    internal var animations = arrayOf(
        "Fade In",
        "Fade Out",
        "Zoom In",
        "Zoom Out",
        "Blink",
        "Rotate",
        "Move",
        "Slide Up",
        "Slide Down",
        "Bounce"
    )
    internal var animationIDs = intArrayOf(
        R.anim.fade_in,
        R.anim.fade_out,
        R.anim.zoom_in,
        R.anim.zoom_out,
        R.anim.blink,
        R.anim.rotate,
        R.anim.move,
        R.anim.slide_up,
        R.anim.slide_down,
        R.anim.bounce
    )

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_animacao)

        lv.adapter = ArrayAdapter(this,
android.R.layout.simple_list_item_1, animations)

        lv.onItemClickListener = AdapterView.OnItemClickListener {
parent, view, position, id ->

            val animation = AnimationUtils.loadAnimation(
                this,
                animationIDs[position]
            )
```

```
                tv.startAnimation(animation)
            }
        }
}
```

Código-fonte 6.33 – Classe AnimacaoActivity
Fonte: Elaborado pelo autor (2020)

Se você chegou até aqui, com certeza tem um aplicativo completo com múltiplas telas, recursos audiovisuais e tudo o que uma aplicação estática oferece ao usuário.

Nosso próximo passo é como integrar os componentes que já vimos até aqui com APIs externas.

REFERÊNCIAS

GUERRA, A. E. **O primeiro mapa-múndi**. Ciência Hoje. 2011. Disponível em: <http://cienciahoje.org.br/artigo/o-primeiro-mapa-mundi/>. Acesso em: 07 out. 2020.